Designing for GitOps and ChatOps

Chris B. Behrens
SOFTWARE ARCHITECT
@chrisbbehrens

A Massive Jenkins Client

A big Jenkins build client

Classic builds

Rudimentary version control

And they can be rendered as code…

But they weren’t proper pipeline scripts

With Groovy script

Introducing New Libraries

- Code agnostic
- Branch agnostic
- Version control agnostic
- The build compiles and packages specific libraries
- So, it can't build a branch with a package not already in the build
- At best, nothing happens...
- At worst, it breaks
- So, we just modify the build
- Now, the build is broken for ALL other branches
- The developer can build the code...
- But nobody else can

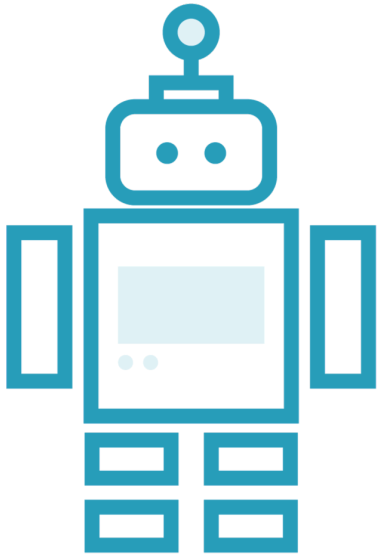
The Solution

A build definition that flows alongside the code

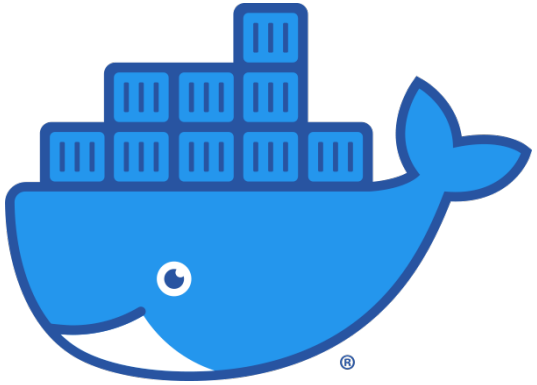
The new build definition is on the same branch as the new code

A pull request reconciles all of this together once and for all

Infrastructure as Code in This Paradigm



A deployment scenario



All in Docker containers

The environment should bind to the code state

Otherwise, you'll end in tears

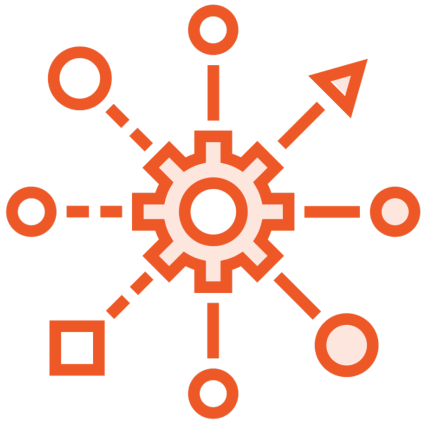
Operations by Pull Request

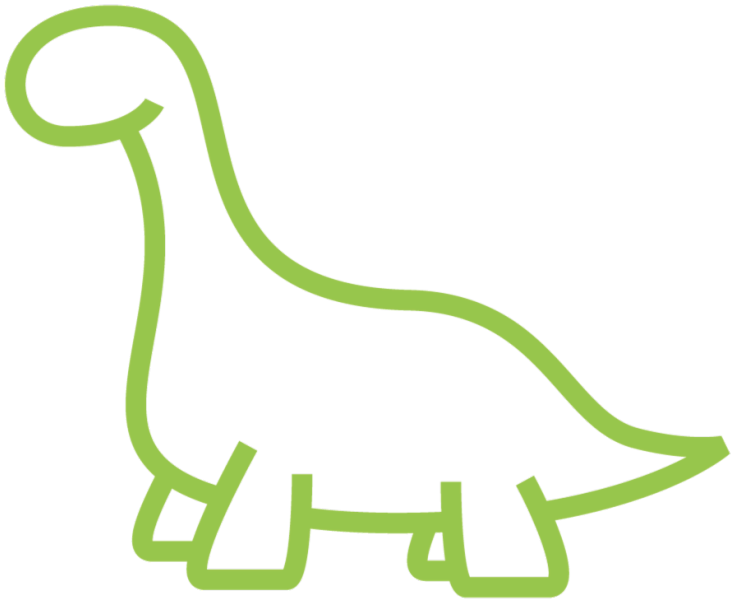
The infrastructure definition should flow alongside the code

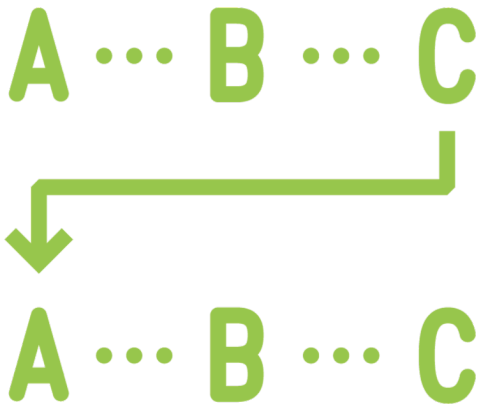
But builds are discrete

Infrastructure is continuous

How an Organization Manages Infrastructure







Recovering from This

"Document it next time"

But documentation is usually prescriptive instead of declarative

Moving Past Scripts

Scripts are better than nothing

SQL – Structured Query Language

INSERT and UPDATE

UPSERT – both operations together

“I want there to be a record with an ID of 2112, the first name Chris, last name Behrens, and the state is Texas.”

Idempotence

Scripts are not idempotent unless intentionally made that way

To make it idempotent

- You have to examine the state before acting
- Whether the resource is present
- And whether it is configured correctly
- Given those answers, you remediate

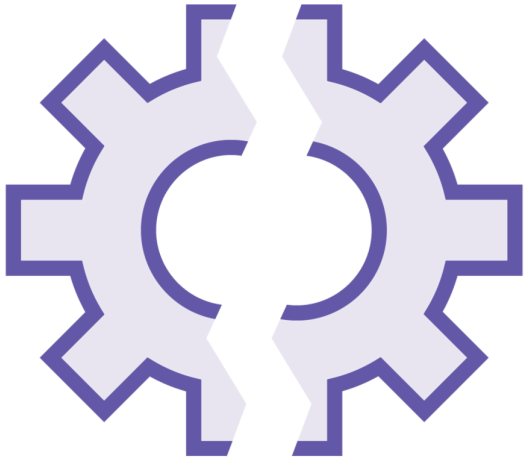
If the service should be absent

- However that is represented
- It should be removed

Idempotence means “having the same effect every time”

Configuration Drift

A desired configuration



Not reflected in reality

Things got off track somewhere...

Usually because of some*one*

Configuration Drift as an Attack

People installing stuff on your stuff

A default Sql Server install

A mystery web server

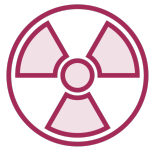
Attackers had compromised my Sql Server

Which allowed them to take control

Security Is the Key

The answer to both configuration drift vectors is security

No way to manually adjust config



No remote admin

“But I need to make adjustments”

Fix the lag time, not the permissions

Managing Configuration Drift

Desired State Configuration

Kubernetes manages containers' config drift

Powershell Desired State Configuration

DSC ensures the configuration state of virtual machines

DSC is declarative rather than programmatic

webdsc.ps1

```
Configuration WebsiteTest {

    # Import the module that contains the resources we're using.
    Import-DscResource -ModuleName PsDesiredStateConfiguration

    # The Node statement specifies which targets this configuration will be applied to.
    Node 'localhost' {

        # The first resource block ensures that the Web-Server (IIS) feature is enabled.
        WindowsFeature WebServer {
            Ensure = "Present"
            Name   = "Web-Server"
        }

        # The second resource block ensures that the website content copied to the website root folder.
        File WebsiteContent {
            Ensure = 'Present'
            SourcePath = 'c:\test\index.htm'
            DestinationPath = 'c:\inetpub\wwwroot'
        }
    }
}
WebsiteTest --Output /output/
```

Putting Our Configuration into Action

```
WebsiteTest --Output /output/
```

Then you publish it to the Local Configuration Manager for the VM

How This Works with GitOps

Your Powershell DSC exists in Git

To change it, you branch…

- Make the changes
- Verify it locally
- Issue a pull request
- After review, the change is merged
- The merge triggers a build
- Which publishes the changes to the LCM
- Which enforces the changes

The best possible history – VCS history

Built-in review

Extensive logs

With work items associated, you have traceability

The who, when, what and why of your GitOps

ARM Templates and GitOps

Azure Resource Manager (ARM) templates

This IS IaC

But not GitOps

Because there's no automatic remediation

What is ChatOps?

The Centrality of Chat

We're not talking about PhoneOps

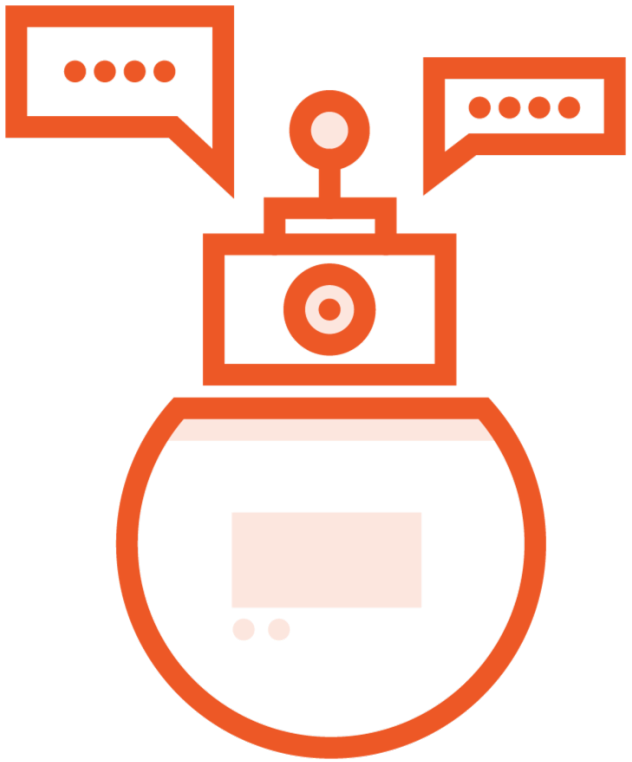
Or EmailOps

People pay closer attention to chat

Chat driving processes is a good thing

Processes driving chat is even better

The Bot



A piece, or pieces of software which delivers automated messaging

Ideally, the bot works in both directions

Different channels for different areas of communication

http://www.pluralsight.com/courses/microsoft-devops-solutions-automating-communication

Failed build notifications

Quick and easy to create

The bare minimum to call it ChatOps

ChatOps, End to End

Failure notification is only ChatOps 101

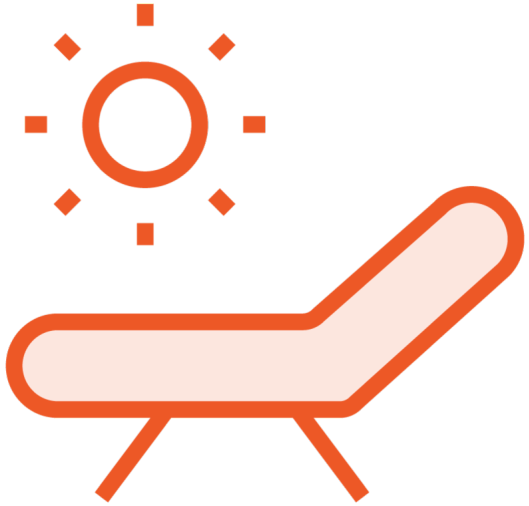
"put tools at the center of the conversation"

Build failure should only be a small part of your process

Implementing a Comprehensive ChatOps Sequence

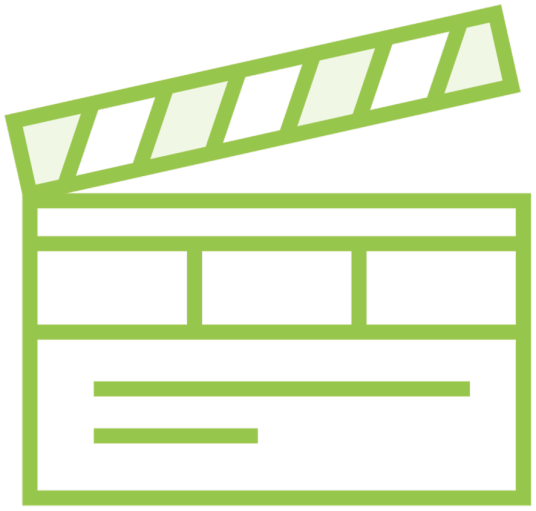
It all starts with a story

ChatOps has nothing to do here



Too early for action

Action Item ChatOps

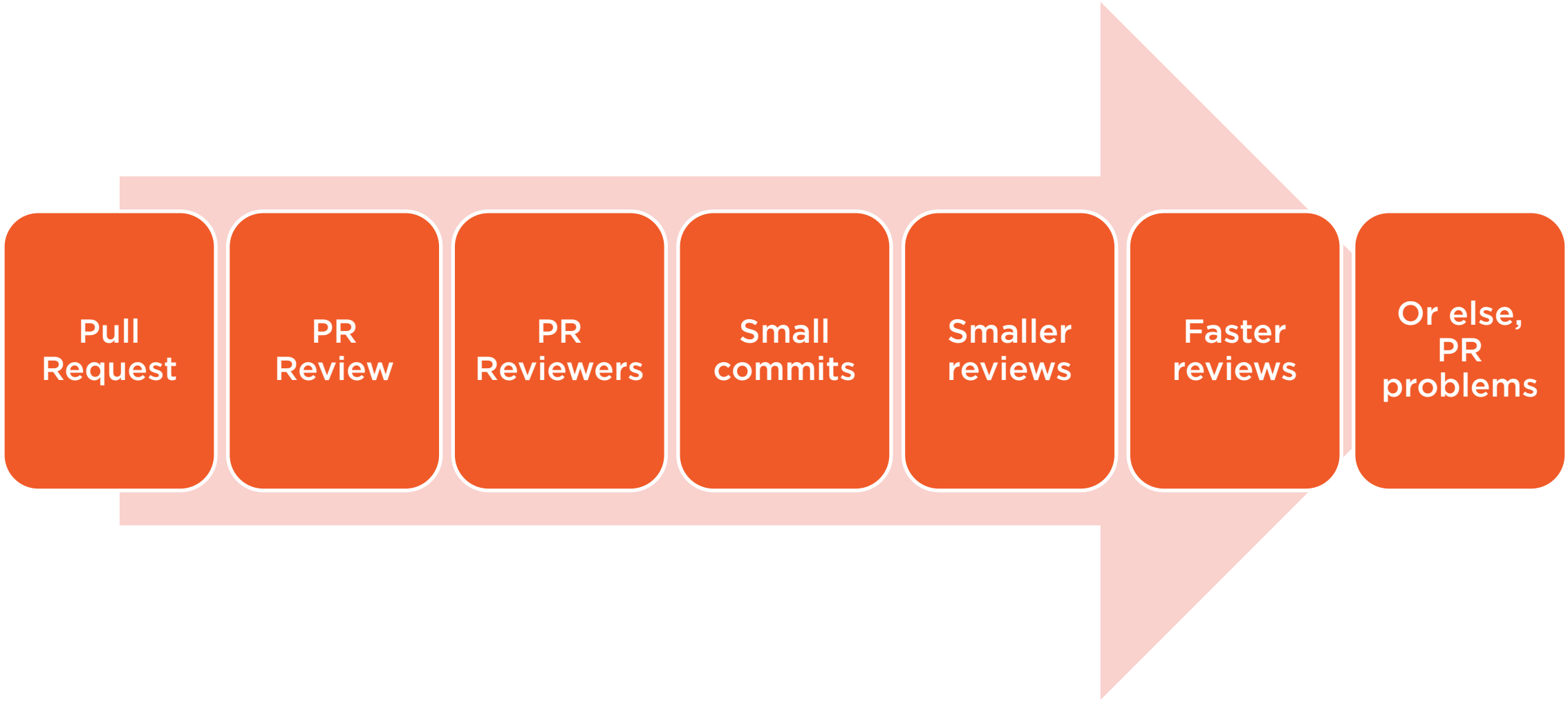


Action Items can be the start of your ChatOps

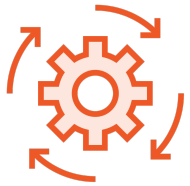
Automated messaging for Work Items

Notify testers of availability

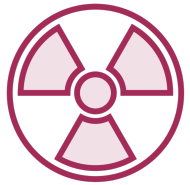
Pull Request ChatOps



Build Status ChatOps



After an approved PR, a build



Some builds are so important, you even care when they succeed

Build status, with a link to the commit that triggered it

Logic Apps

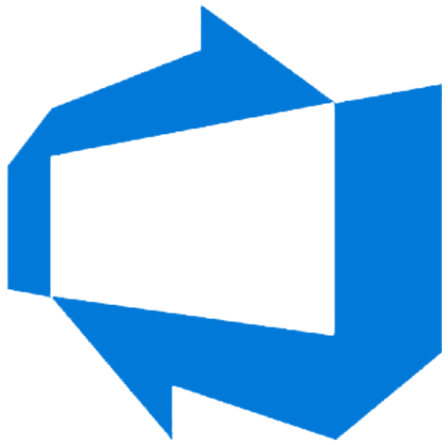
One more exam concept

Logic Apps glue stuff together

With more complex filtering and dispatching

A good choice for event-based action

Integrating an Azure Alert with a Logic App



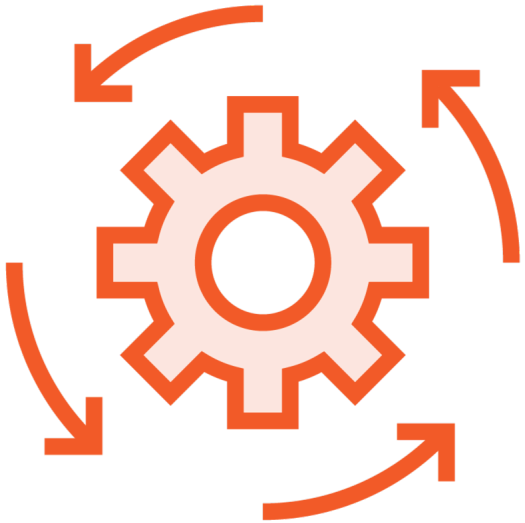
https://bit.ly/3g1BSd9

1. Deploy the template to Azure (by clicking a button on the page I linked)
2. Establish the API connection to Slack
3. Create an alert which monitors your application for a condition
4. Associate that alert with your Logic App

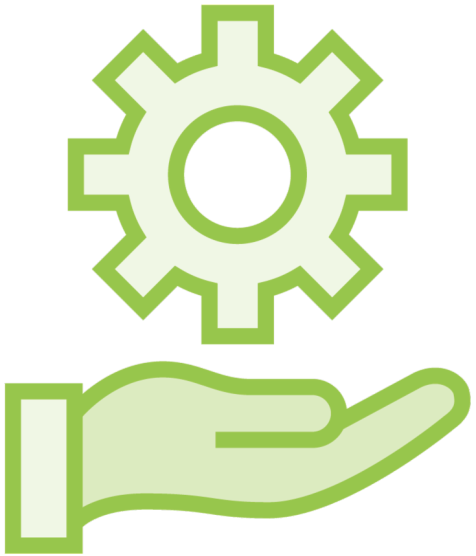
Keep communication actionable.

What Happens Next

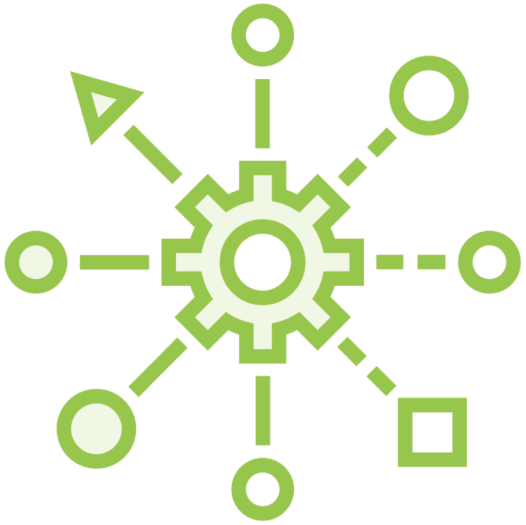
The operator knows that action is needed



Logic Apps can run in the other direction, too



If your processes are sufficiently mature...



You could take an application live inside your Chat tool

Course Summary

Integrating with GitHub

GitOps Basics

- Configuration drift
- Desired State Configuration

ChatOps

- What it is
- Moving from a minimal implementation
- An end-to-end implementation
- Implementing ChatOps with Azure DevOps

THANK YOU FOR WATCHING!!!